CSE 591 Randomized and Approximation Algorithms

Goran Konjevod

Department of Computer Science and Engineering
Arizona State University

ASU, Fall 2008

Logistics 1

- Instructor: Goran Konjevod, BY450, goran@asu.edu
- Office Hours: TTh 13:30–15:00
- TA: Melih Onus, monus@asu.edu
- Office Hours: TBA

Logistics 2: Evaluation

- Homeworks (6–7): 30%
- Midterm (early November): 30%
- Project: 40%

Randomization

Leaving decisions to chance.

Randomized algorithm: allowed to invoke a random event and use the outcome to determine the next step.

Basic random events:

1. Basic: generate a random bit
2. Complex: generate a random number (int/float)
3. Complex: generate a random object of some general type

Uses of randomization

Randomization may

- make complicated algorithms simpler
- make inefficient computations efficient (quicksort, mincut)
- make possible things we don't know how to do deterministically (~~primality testing in P~~, matching in parallel)
- make possible things that are provably impossible to do deterministically (volume computation, distributed protocols)

Example: quicksort

Input: set S of numbers.
Output: the elements of S sorted in increasing order.

1. Choose $y \in S$ uniformly at random.
2. $S_1 = \{x \in S \mid x < y\}$, $S_2 = \{x \in S \mid x > y\}$.
3. Recursively sort S_1 and S_2.
4. Output sorted S_1, followed by y, followed by sorted S_2.

Randomized quicksort analysis

Let $s_1 \leq s_2 \leq \ldots \leq s_n$ be the set S in order.
Let
$$X_{ij} = \begin{cases} 1 & s_i \text{ is compared to } s_j \\ 0 & s_i \text{ is not compared to } s_j \end{cases}$$
Number of comparisons made is $T_n = \sum_{i=1}^{n} \sum_{j>i} X_{ij}$.

Digression: discrete probability (1)

Sample space: set Ω of all possible outcomes (quicksort: the set of all possible runs of the algorithm on input S)
Events: subsets of Ω (example: let x be an element of S. The set A_x of all possible runs where the first element selected is x)
Probability of an event ($\Pr[A_x] = 1/n$).
Random variable: mapping from Ω to real numbers (X_{ij}).
Expectation of a random variable: its "average value",

$$\mathbf{E}[X] = \sum_{\omega \in \Omega} \Pr[\omega] \cdot X(\omega)$$

($\mathbf{E}[X_{ij}] = \Pr[X_{ij} = 1] \cdot 1 + \Pr[X_{ij} = 0] \cdot 0 = \Pr[X_{ij} = 1]$).

Digression: discrete probability (2)

Expectation is linear: for random variables X and Y, numbers a, b,

$$\mathbf{E}[aX + bY] = a\mathbf{E}[X] + b\mathbf{E}[Y].$$

Example: in randomized quicksort,

$$\begin{aligned}\mathbf{E}[T_n] &= \mathbf{E}[\sum_{i=1}^{n}\sum_{j>i}[X_{ij}] \\ &= \sum_{i=1}^{n}\sum_{j>i}\mathbf{E}[X_{ij}] \\ &= \sum_{i=1}^{n}\sum_{j>i}\Pr[X_{ij} = 1].\end{aligned}$$

Randomized quicksort analysis

$X_{ij} = 1$ if and only if s_i and s_j are compared.
When are s_i and s_j compared?

Exactly if either s_i or s_j is selected before any of the elements $s_i, s_{i+1}, \ldots, s_{j-1}, s_j$.

The probability of this happening is $2/(j - i + 1)$.

Randomized quicksort analysis

$$\begin{aligned}
\mathbf{E}[T_n] &= \sum_{i=1}^{n} \sum_{j>i} \Pr[X_{ij} = 1] = \sum_{i=1}^{n} \sum_{j>i} \frac{2}{j-i+1} \\
&\le \sum_{i=1}^{n} \sum_{k=1}^{n-i+1} \frac{2}{k} \\
&\le 2 \sum_{i=1}^{n} \sum_{k=1}^{n} \frac{1}{k} \le 2nH_n = O(n \ln n).
\end{aligned}$$

Cuts in graphs

A *cut* in G: a set of edges that disconnects the graph.
For a set $C \subseteq V$, let $\overline{C} = V \setminus C$. Then $(C, \overline{C})$ defines a cut. We write

$$(C, \overline{C}) = \{uv \in E \mid u \in C,\ v \in \overline{C}\}.$$

Minimum cut problem

Input: (multi)graph $G = (V, E)$.
Output: a cut of minimum cardinality in G.

Polynomially solvable, $O(n^3)$ time (but not simple). Need n minimum st-cuts or the Stoer-Wagner algorithm.

Simple randomized algorithm for mincut

1. Pick an edge e uniformly at random.
2. Contract e.
3. Repeat until there are only two vertices left.

Randomized mincut analysis (1)

Claim: Contractions do not decrease the minimum cut value.

Randomized mincut analysis (2)

Let k be the minimum cut cardinality. Let C be a minimum cut.

G has at least $kn/2$ edges.

For $i = 1, \ldots, n-2$, let A_i be the event that no edge of C was contracted in the i-th step.

If all of the events $A_1, \ldots A_{n-2}$ happen, then the algorithm finds the minimum cut C.

Randomized mincut analysis (3)

$$\Pr[A_1] \geq 1 - \frac{2}{n} = \frac{n-2}{n}.$$

If A_1 happens, then before the second step of the algorithm there are at least $k(n-1)/2$ edges in the graph.

$$\Pr[A_2 \mid A_1] \geq 1 - \frac{2}{n-1} = \frac{n-3}{n-1}.$$

In general, if $A_1, \ldots, A_{i-1}$ happen, then before the i-th step there are at least $k(n-i+1)/2$ edges in the graph and so

$$\Pr[A_i \mid A_1, A_2, \ldots A_{i-1}] \geq 1 - \frac{2}{n-i+1} = \frac{n-i-1}{n-i+1}.$$

Randomized mincut analysis (4)

The probability that no edge of C is contracted is

$$\Pr[A_1 \cap A_2 \cap \cdots \cap A_{n-2}]$$

Digression: discrete probability (3)

Two events A and B are *independent*, if

$$\Pr[A \cap B] = \Pr[A] \Pr[B].$$

The conditional probability of A given B is defined by

$$\Pr[A \mid B] = \frac{\Pr[A \cap B]}{\Pr[B]}.$$

For any two events A, B, we have

$$\Pr[A \cap B] = \Pr[A \mid B] \cdot \Pr[B].$$

In general,

$$\Pr[A_1 \cap A_2 \cap \cdots \cap A_k] = \Pr[A_k \mid A_1 \cap \cdots \cap A_{k-1}] \cdots \Pr[A_2 \mid A_1] \cdot \Pr[A_1].$$

Randomized mincut analysis (5)

The probability that no edge of C is contracted is

$$\begin{aligned}\Pr[A_1 \cap A_2 \cap \cdots \cap A_{n-2}] &\geq \frac{n-2}{n} \cdot \frac{n-3}{n-1} \cdot \frac{n-4}{n-2} \cdots \frac{3}{5} \cdot \frac{2}{4} \cdot \frac{1}{3} \\ &= \frac{2}{n(n-1)}.\end{aligned}$$

Randomized mincut analysis (6)

The probability that C is output by the algorithm is at least $2/n^2$.

Suppose we repeat the algorithm $n^2/2$ times, each time with new independent random choices.

The probability that C is not found in any of the $n^2/2$ runs is then at most

$$(1 - \frac{2}{n^2})^{n^2/2} < \frac{1}{e}.$$

Improved randomized minimum cut (1)

So far: an $O(n^2 m)$ algorithm for mincut.
To improve, notice that earlier steps are safer than later ones.

How far can we go until the probability of having lost C is $1/2$?
If there are about $n/\sqrt{2}$ vertices left, the success probability is at least

$$\frac{n-2}{n} \cdot \frac{n-3}{n-1} \cdots \frac{n/\sqrt{2}-2}{n/\sqrt{2}} = \frac{(n/\sqrt{2}-3)(n/\sqrt{2}-2)}{n(n-1)},$$

that is, roughly

$$\frac{1}{2}.$$

Improved randomized minimum cut (2)

Now think of these first $n - n/\sqrt{2}$ steps as a single experiment! Its outcome is either success or failure, and the probability of success is at least $1/2$.

Perform this experiment twice, then if one of the two runs succeeded, recurse.

Build a binary tree to describe the process.
Depth $= 2 \lg n$, number of leaves $= n^2$.
If each edge is erased independently with probability $1/2$, what is the probability that a root-leaf path survives?

Improved randomized minimum cut (3)

If P_d is the probability a path survives in a tree of depth d, then

$$\begin{aligned} P_d &= \frac{1}{2}P_{d-1} + \frac{1}{4}(1-(1-P_{d-1})^2) \\ &= \frac{1}{2}P_{d-1} + \frac{1}{4}(2P_{d-1}-(P_{d-1})^2) \\ &= P_{d-1} - \frac{1}{4}(P_{d-1})^2. \end{aligned}$$

Now if $P_{d-1} > \frac{1}{d-1}$, then

$$P_d > \frac{1}{d-1} - \frac{1}{4(d-1)^2} > \frac{1}{d-1} - \frac{1}{d(d-1)} = \frac{1}{d}.$$

So the probability a path survives is $\Omega(\frac{1}{\log n})$.
To make this a constant, repeat independently $\log n$ times.

Improved randomized minimum cut (4)

What is the running time of a single "tree" process?
$T(n) = O(n^2) + 2T(n/\sqrt{2}) = O(n^2 \lg n)$.
The total running time of the improved version is then $O(n^2 \lg^2 n)$.